C interfaces to GALAHAD IR

Jari Fowkes and Nick Gould
STFC Rutherford Appleton Laboratory

Thu Jun 22 2023

Chapter 1

GALAHAD C package ir

1.1 Introduction

1.1.1 Purpose

Given a sparse symmetric $n \times n$ matrix $A = a_{ij}$ and the factorization of A found by the GALAHAD package SLS, this package **solves the system of linear equations** $Ax = b$ **using iterative refinement.**

Currently, only the control and inform parameters are exposed; these are provided and used by other GALAHAD packages with C interfaces.

1.1.2 Authors

N. I. M. Gould, STFC-Rutherford Appleton Laboratory, England.

C interface, additionally J. Fowkes, STFC-Rutherford Appleton Laboratory.

Julia interface, additionally A. Montoison and D. Orban, Polytechnique Montréal.

1.1.3 Originally released

October 2008, C interface January 2022

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

File Documentation

3.1 galahad_ir.h File Reference

```
#include <stdbool.h>
#include <stdint.h>
#include "galahad_precision.h"
#include "galahad_cfunctions.h"
```

Data Structures

- struct ir_control_type
- struct ir_inform_type

Functions

- void ir_initialize (void ∗∗data, struct ir_control_type ∗control, int ∗status)
- void ir_information (void ∗∗data, struct ir_inform_type ∗inform, int ∗status)
- void ir_terminate (void ∗∗data, struct ir_control_type ∗control, struct ir_inform_type ∗inform)

3.1.1 Data Structure Documentation

3.1.1.1 struct ir_control_type

control derived type as a C struct

Data Fields

bool	f_indexing	use C or Fortran sparse matrix indexing
int	error	unit for error messages
int	out	unit for monitor output
int	print_level	controls level of diagnostic output
int	itref_max	maximum number of iterative refinements allowed

Data Fields

real_wp_	acceptable_residual_relative	refinement will cease as soon as the residual $\|\|Ax-b\|\|$ falls below max(acceptable_residual_relative $*$ $\|\|b\|\|$, acceptable_residual_absolute)
real_wp_	acceptable_residual_absolute	see acceptable_residual_relative
real_wp_	required_residual_relative	refinement will be judged to have failed if the residual $\|\|Ax-b\|\| \geq$ required_residual_relative $*$ $\|\|b\|\|$. No checking if required_residual_relative < 0
bool	record_residuals	record the initial and final residual
bool	space_critical	if space is critical, ensure allocated arrays are no bigger than needed
bool	deallocate_error_fatal	exit if any deallocation fails
char	prefix[31]	all output lines will be prefixed by prefix(2:LEN(TRIM(.prefix))-1) where prefix contains the required string enclosed in quotes, e.g. "string" or 'string'

3.1.1.2 struct ir_inform_type

inform derived type as a C struct

Data Fields

int	status	the return status. Possible values are: • 0 the solution has been found. • -1. An allocation error occurred. A message indicating the offending array is written on unit control.error, and the returned allocation status and a string containing the name of the offending array are held in inform.alloc_status and inform.bad_alloc respectively. • -2. A deallocation error occurred. A message indicating the offending array is written on unit control.error and the returned allocation status and a string containing the name of the offending array are held in inform.alloc_status and inform.bad_alloc respectively. • -11. Iterative refinement has not reduced the relative residual by more than control.required_relative_residual.
int	alloc_status	the status of the last attempted allocation/deallocation.
char	bad_alloc[81]	the name of the array for which an allocation/deallocation error occurred.
real_wp_	norm_initial_residual	the infinity norm of the initial residual
real_wp_	norm_final_residual	the infinity norm of the final residual

3.1.2 Function Documentation

3.1.2.1 ir_initialize()

```
void ir_initialize (
             void ** data,
             struct ir_control_type * control,
             int * status )
```

Set default control values and initialize private data

Parameters

in,out	*data*	holds private internal data
out	*control*	is a struct containing control information (see ir_control_type)
out	*status*	is a scalar variable of type int, that gives the exit status from the package. Possible values are (currently): • 0. The initialization was succesful.

3.1.2.2 ir_information()

```
void ir_information (
             void ** data,
             struct ir_inform_type * inform,
             int * status )
```

Provides output information

Parameters

in,out	*data*	holds private internal data
out	*inform*	is a struct containing output information (see ir_inform_type)
out	*status*	is a scalar variable of type int, that gives the exit status from the package. Possible values are (currently): • 0. The values were recorded succesfully

3.1.2.3 ir_terminate()

```
void ir_terminate (
             void ** data,
             struct ir_control_type * control,
             struct ir_inform_type * inform )
```

Deallocate all internal private storage

Parameters

`in,out`	*data*	holds private internal data
`out`	*control*	is a struct containing control information (see ir_control_type)
`out`	*inform*	is a struct containing output information (see ir_inform_type)